L3: Entry 1 of 137 File: PGPB Jul 11, 2002

DOCUMENT-IDENTIFIER: US 20020089695 A1
TITLE: PRINT CONTROL APPARATUS AND METHOD AND STORAGE MEDIUM

Application Filing Date:
19970827

W.D

CLAIMS:

4. A print control apparatus in which a data dedompressor receives compressed data from a memory and decompresses said data and outputs the decompressed data to a recorder, wherein said data decompressor comprises: a first buffer memory for temporarily holding and outputting the compressed data; a first buffer memory controller for inhibiting an input of the data to said first buffer memory when said first buffer memory is full and for resuming the input of the data to the first buffer memory when the full state of the first buffer memory is released; the data decompressor to which the data outputted from said first buffer memory is inputted and which decompresses said data and outputs; a second buffer memory for temporarily holding the data outputted from said data decompressor and outputting said data to the recorder; and data decompressor control means for detecting an output of the data from said data decompressor to said second buffer memory and outputting a data full signal indicating that said data decompressor is full to the data decompressor, and said data decompressor inhibits the output of the data to said second buffer memory when the data full signal is received from said data decompressor control means and resumes the output of the data to said second buffer memory when the data full signal is released.

L3: Entry 5 of 137 File: USPT Aug 10, 2004

DOCUMENT-IDENTIFIER: US 6775698 B1
TITLE: Apparatus and method for downloading core file in a network device

Application Filing Date (1):
20001229

NO

Detailed Description Text (17):
After the temporary buffer is downloaded into flash memory in step 84, or if the temporary buffer is not full in decision step 82, decision step 86 determines whether there are any more bytes in the main memory DRAM 18. If all bytes of the main memory 18 have been compressed, any remaining compressed data in the temporary buffer is downloaded into local flash memory in decision step 88. If there is more data in main memory 18, decision step 86 reads the next byte in step 78.

L3: Entry 51 of 137 File: USPT Dec 7, 1999

DOCUMENT-IDENTIFIER: US 5999709 A
TITLE: Printer memory boost

Abstract Text (1):
A method and apparatus for allocating memory usage in a visual output device having a render engine, a display list memory for storing display list objects and a compressed band buffer memory for storing compressed bitmap data rendered by the render engine. The method includes storing display list objects in the display list memory and detecting that the display list memory is full. Upon detecting the display list memory is full and determining that the compressed band buffer memory is not full, display list data is transferred from display list memory to the compressed band buffer memory, thereby freeing space in the display list memory space for more display list objects.

NO

Application Filing Date (1):
19970418

Brief Summary Text (9):
In one aspect the invention provides a method for allocating memory usage in a visual output device having a render engine, a display list memory for storing display list objects and a compressed band buffer memory for storing compressed bitmap data rendered by the render engine. The method includes storing display list objects in the display list memory and detecting that the display list memory is full. Upon detecting the display list memory is full and determining that the compressed band buffer memory is not full, display list data is transferred from display list memory to the compressed band buffer memory, thereby freeing space in the display list memory space for more display list objects.

Brief Summary Text (18):
In another aspect, the invention provides a printer configured to receive a page description language description of a page to be printed. The printer includes an interpreter connected to receive page description language input and operating to generate display list objects and cause display list objects to be stored in a display list memory, a rendering engine coupled to the display list memory and operating to receive display list objects stored in the display list memory and to render the objects into an uncompressed band buffer memory. The uncompressed band buffer memory is a memory for storing in uncompressed form at least one band of rendered data. The printer includes a data compression engine coupled to the uncompressed band buffer memory and operating to receive and compress rendered data from the uncompressed band buffer memory and to store compressed data in a compressed band buffer memory. The compressed band buffer memory is sufficiently large to store any full page in some compressed form. Finally, the printer includes a memory manager connected to receive a signal indicating that the display list memory is full. In response to the signal, the memory manager transfers display lists from the display list memory to the compressed band buffer memory unless the compressed band buffer memory is also full.

Brief Summary Text (19):
In another aspect the invention provides a method for allocating memory usage in a

visual output device having a render engine, a display list memory for storing display list objects and a compressed band buffer memory for storing compressed bitmap data rendered by the render engine. The method includes storing display list objects in the display list memory and detecting that the display list memory is full. Upon detecting the display list memory is full and determining that the compressed band buffer memory is not full, display list data is allocated from display list memory to the compressed band buffer memory, thereby freeing space in the display list memory space for more display list objects.

Detailed Description Text (13):
It may occur that the display list memory 60 becomes full and the memory manager 58 finds the compressed band buffer full and therefore unavailable to store display list data. Whether a memory is deemed full may be determined on the basis of a variety of conditions. For example, the compressed band buffer 64 may be deemed full when it has insufficient space to store any of the display list data currently stored in display list memory 60 and still guarantee that the page can be printed successfully. The consolidated memory may be deemed full when it has insufficient space to store another display list object.

Detailed Description Text (14):
When both display list and compressed buffer memories are full, as described, a cycling process is initiated in which display lists for one or more bands are rendered. Decompression engine 56 determines if any data in compressed band buffer 64 corresponds to a band to be rendered in the cycling process. If so, decompression engine 56 decompresses the compressed bitmaps and stores them in a band in uncompressed band buffer 62. Thereafter, render engine 52 renders all display lists associated with the same band, whether stored in the display list memory or in the compressed band buffer, into the band data (if any) in the uncompressed band buffer 62, resulting in a single bitmap for a given band. Subsequently, the band bitmap may be compressed and stored in the compressed band buffer.

Detailed Description Text (24):
In the event that the display list memory is full and the compressed buffer memory is also full as indicated at step 112, then a cycling process is initiated at step 116 resulting in the freeing of space in the display list memory by rendering the display list data stored therein. Cycling is only initiated when both the display list memory and the compressed band buffer are full, thereby minimizing the frequency of cycling operations. By minimizing cycling operations, unnecessary compression may also be avoided.

CLAIMS:

1. A method for allocating memory usage in a visual output device having a render engine, a display list memory for storing display list objects and a compressed band buffer memory for storing compressed bitmap data rendered by the render engine, the method comprising:

storing display list objects in the display list memory; and

detecting that the display list memory is full and upon detecting such and determining that the compressed band buffer memory is not full, transferring one or more of the display list objects from the display list memory to the compressed band buffer memory without rendering and without compressing the display list objects thereby freeing space in the display list memory for more display list objects.

16. A printer configured to receive a page description language description of a page to be printed, comprising:

an interpreter connected to receive a page description language input and operating to generate display list objects and cause the display list objects to be stored in a display list memory;

a rendering engine coupled to the display list memory and operating to receive the display list objects stored in the display list memory and to render the objects into an uncompressed band buffer memory, being a memory for storing at least one band of rendered data in uncompressed form;

a data compression engine coupled to the uncompressed band buffer memory and operating to receive and compress rendered data from the uncompressed band buffer memory and to store compressed data in a compressed band buffer memory, the compressed band buffer memory being sufficiently large to store any full page in some compressed form; and

a memory manager connected to receive a signal indicating that the display list memory is full and operating in response to the signal to transfer one or more of the display list objects from the display list memory to the compressed band buffer memory without rendering and without compressing the display list objects unless the compressed band buffer memory is also full.

17. A method for allocating memory usage in a visual output device having a render engine, a display list memory for storing display list objects and a compressed band buffer memory for storing compressed bitmap data rendered by the render engine, the method comprising:

storing display list objects in the display list memory;

allocating space in the compressed band buffer memory to store the display list objects;

detecting that the display list memory is full and upon detecting such and determining that the compressed band buffer memory is not full, transferring display list obiects from display list memory to the compressed band buffer memory without rendering and without compressing the display list objects, thereby freeing space in the display list memory for more display list objects.

Previous Doc Next Doc Go to Doc#

L3: Entry 12 of 137 File: USPT Dec 2, 2003

DOCUMENT-IDENTIFIER: US 6658027 B1
TITLE: Jitter buffer management

Application Filing Date (1):
19990816

Detailed Description Text (21):
Returning to step 450, if the jitter buffer is not empty (which would be the normal state during a call), the processor adds a frame to the jitter buffer 480. The processor then evaluates the condition of the buffer. In this example, the processor determines whether the jitter buffer is full beyond the high water mark level 485. If the answer is negative, the processor simply returns to the wait-for-next-frame step 400. However, if the jitter buffer is full beyond the high water mark level, the processor determines whether the RemoveFrameCount variable equals zero 490, and if not, it returns to the wait-for-next-frame step. However, if the RemoveFrameCount is equal to zero, the frame input process sets the RemoveFrameCount variable to a number representing the number of frames the system will need to delete from the buffer in order to reach the desired jitter level. In this embodiment, this value is determined to be the high water mark minus the desired jitter level as shown at 495 and then returns to the wait-for-next-frame step.

WK

L3: Entry 75 of 137 File: USPT Mar 31, 1998

DOCUMENT-IDENTIFIER: US 5734748 A
**** See image for Certificate of Correction ****
TITLE: Character pattern generation apparatus and method

Application Filing Date (1):
19941223

Detailed Description Text (134):
Referring back to FIG. 26, if it is determined in step (12) that the cache condition is satisfied, the flow advances to step (13) to cache data. The flow then advances to step (14), and the generated coordinate data (control point data) in the outline format of the object stroke or the coordinate data (control point data) in the outline format got in step (8) is shifted to the development position in an object character element. The flow advances to step (15) to check if the above-mentioned operation has been completed for all strokes constituting the object character element. If N in step (15), the flow returns to the entrance of step (7) to perform the same operation for the next object stroke. However, if Y in step (15), the flow advances to step (16) to check if this stroke satisfies the cache condition. The cache condition test flow is shown in FIG. 5, and the cache buffer state at that time is shown in FIGS. 6A to 6C. In step (1) in FIG. 5, it is checked if the buffer for storing the cached characters or character elements is already full of data. If the buffer is full of data, it is checked in step (2) in FIG. 5 if the priority of an object character or character element is higher than that of data stored in the cache buffer. If data with priority lower than that of the object character or character element is not stored in the cache buffer, the object character or character element is not cached. However, if at least one data with priority lower than that of the object character or character element is stored in the cache buffer, data with the lowest priority is deleted from the cache buffer, and the data of the object character or character element is stored in the cache buffer. Referring back to FIG. 26B, if the cache condition is satisfied, the flow advances to step (17) to cache data. The flow then advances to step (18), and the generated coordinate data (control point data) in the outline format by stroke synthesis or the coordinate data (control point data) in the outline format got in step (5) is shifted to the development position in an object character. In step (19), it is checked if the above-mentioned operation has been completed for all character elements constituting the character. If N in step (19), the flow returns to the entrance of step (4) to perform the same operation for the next object character element. However, if Y in step (19), it is checked in step (20) if this character satisfies the cache condition, as in step (16). If the cache condition is satisfied, the flow advances to step (21) to cache data. The flow then advances to step (22) to check if control point data is straight data, so as to interpolate an object stroke. If control point data is not straight data, curve data is determined, and interpolation is performed by a predetermined curve interpolation method in step (23). Thereafter, dots are written in step (24). If it is determined in step (22) that control point data is straight data, dots are written in step (25). It is then checked in step (26) if all dots for one outline of data of the object stroke have been written. If N in step (26), the flow returns to the entrance of step (22); otherwise, the flow advances to step (27) to check if the interpolation process has been completed for all strokes constituting the

character. If N in step (27), the flow returns to the entrance of step (22); otherwise, the flow advances to step (28) to perform painting.

Detailed Description Text (137):
Referring back to FIG. 28B, if it is determined in step (19) that the cache condition is satisfied, the flow advances to step (20) to cache data. The flow advances to step (21) to check if bit map generation and synthesis of all strokes constituting the object character element have been completed. If N in step (21), the flow returns to the entrance of step (7) to perform the same operation for the next object stroke. However, if Y in step (21), the flow advances to step (22) to check if this character element satisfies the cache condition. The cache condition test flow is shown in FIG. 5, and the cache buffer state at that time is shown in FIGS. 6A to 6C. In step (1) in FIG. 5, it is checked if the buffer for storing the cached characters or character elements is already full of data. If the buffer is full of data, it is checked in step (2) in FIG. 5 if the priority of an object character or character element is higher than that of data stored in the cache buffer. If data with priority lower than that of the object character or character element is not stored in the cache buffer, the object character or character element is not cached. However, if at least one data with priority lower than that of the object character or character element is stored in the cache buffer, data with the lowest priority is deleted from the cache buffer, and the data of the object character or character element is stored in the cache buffer. Referring back to FIG. 28B, if it is determined in step (22) that the cache condition is satisfied, the flow advances to step (23) to cache data. It is then checked in step (24) if bit map generation and synthesis of all character elements constituting a character have been completed. If N in step (24), the flow returns to the entrance of step (4) to perform the same operation for the next character element. However, if Y in step (24), it is checked in step (25) if this character satisfies the cache condition, as in step (22). If the cache condition is satisfied, the flow advances to step (26) to cache data. In this manner, pattern generation of the object character ends.

L3: Entry 79 of 137 File: USPT Nov 11, 1997

DOCUMENT-IDENTIFIER: US 5687279 A
**** See image for Certificate of Correction ****
TITLE: Retro-storing analog information in a digital storage circuit

Application Filing Date (1):
19941223

Detailed Description Text (41):
Controller 35 also executes a process in accordance with one embodiment of the present invention to store the digital data received from digital signal processor 34 in either buffer 36 or flash memory array 37 in accordance with the STORAGE.sub.-- EN signal. When the STORAGE.sub.-- EN signal is not asserted, controller 35 stores the data received from digital signal processor 34 in buffer 36 in the first-in-first-out manner. The length of FIFO buffer 36 can be selected at user's desire. Controller 35 stores one packet of digital data in buffer 36 at one time. When buffer 36 if full, the earliest stored data packet is deleted from buffer 36. This allows buffer 36 to store data in a continuous FIFO loop. At this time, flash memory array 37 does not store any data received from digital signal processor 34.

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L3: Entry 10 of 137 File: USPT Dec 16, 2003

DOCUMENT-IDENTIFIER: US 6664982 B1
TITLE: Multi-user on-screen keyboard

Application Filing Date (1):
19970115

Detailed Description Text (307):
An important aspect of the invention relates to the manner in which the audio data is compressed. Referring to FIGS. 87 and 88, audio data, prior to being compressed, is stored in a temporary buffer in the wireless interface device 100. Uncompressed data, as illustrated in FIG. 88, is sampled every predetermined time period, or when the volume is below a predetermined level as illustrated and stored in a temporary buffer. As illustrated in FIG. 88, the sample points on the horizontal axes marked with the `X` are exemplary data points stored in the temporary buffer. As shown, the points 1 and 2 are at predetermined time intervals, while the point between 2 and 3 seconds is a point where the amplitude or volume is below a predetermined level. Thus, as indicated in step 2064, the system samples the audio data points at every predetermined time period or when the volume has reached a predetermined level and places the data in a temporary buffer in step 2066. The system loops back to step 2064 and continues sampling data points until the buffer is full, as ascertained in step 2068. Once the temporary audio buffer is full, the entire buffer is compressed at one time, as indicated by step 2070. The compressed audio data is then passed to the wireless interface device client manager in order to pass the data over the radio link to the server 1708 in step 2072.

Previous Doc Next Doc Go to Doc#

L3: Entry 8 of 137 File: USPT Apr 20, 2004

DOCUMENT-IDENTIFIER: US 6724372 B1
TITLE: Ink trails on a wireless remote interface tablet and wireless remote ink field object

Application Filing Date (1):
19970115

Detailed Description Text (307):
An important aspect of the invention relates to the manner in which the audio data is compressed. Referring to FIGS. 87 and 88, audio data, prior to being compressed, is stored in a temporary buffer in the wireless interface device 100. Uncompressed data, as illustrated in FIG. 88, is sampled every predetermined time period, or when the volume is below a predetermined level as illustrated and stored in a temporary buffer. As illustrated in FIG. 88, the sample points on the horizontal axes marked with the `X` are exemplary data points stored in the temporary buffer. As shown, the points 1 and 2 are at predetermined time intervals, while the point between 2 and 3 seconds is a point where the amplitude or volume is below a predetermined level. Thus, as indicated in step 2064, the system samples the audio data points at every predetermined time period or when the volume has reached a predetermined level and places the data in a temporary buffer in step 2066. The system loops back to step 2064 and continues sampling data points until the buffer is full, as ascertained in step 2068. Once the temporary audio buffer is full, the entire buffer is compressed at one time, as indicated by step 2070. The compressed audio data is then passed to the wireless interface device client manager in order to pass the data over the radio link to the server 1708 in step 2072.